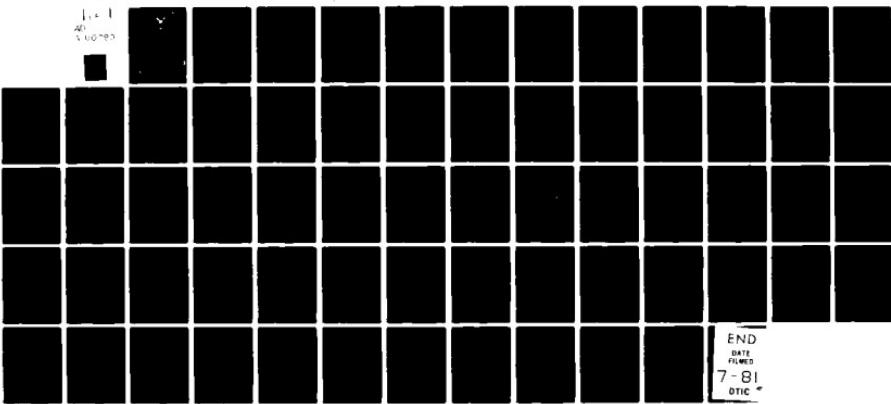


AD-A100 790 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHO0--ETC F/G 9/2  
MICROCODE DEVELOPMENT SYSTEM FOR THE MICROPROGRAMMABLE MINICOMP--ETC(U)  
DEC 80 G H CARROLL

UNCLASSIFIED AFIT/GE/EE/80D-12

NL



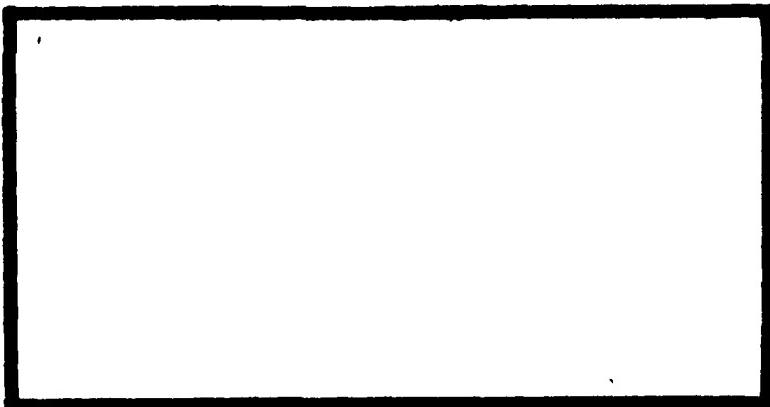
END  
F  
F1  
7-81  
DTIC \*

DOC

LEVEL II

(1)

AD A100790



DTIC FILE COPY

This document has been approved  
for public release and sale; its  
distribution is unlimited.

DEPARTMENT OF THE AIR FORCE  
AIR UNIVERSITY (ATC)  
AIR FORCE INSTITUTE OF TECHNOLOGY

3 DTIC  
ELECTED  
JUL 1 1981  
D

Wright-Patterson Air Force Base, Ohio

816 30 084

(14)

AFIT/GE/EE/80D-12

(1) Dr. 80

(12) 65

(1) Master's thesis,

MICROCODE DEVELOPMENT SYSTEM FOR THE  
MICROPROGRAMMABLE MICROCOMPUTER EMULATOR

(MIME) .

AFIT/GE/EE/80D-12 / Glenn D. Carroll  
Captain USAF

Approved for public release; distribution unlimited.

012225

De

AFIT/GE/EE/80D-12

MICROCODE DEVELOPMENT SYSTEM FOR THE  
MICROPROGRAMMABLE MINICOMPUTER EMULATOR (MIME)

THESIS

Presented to the Faculty of the School of Engineering  
of the Air Force Institute of Technology  
Air University (ATC)  
in Partial Fulfillment of the  
Requirements for the Degree of  
Master of Science

by

Glenn H. Carroll, B.S.E.E.  
Captain USAF  
Graduate Electrical Engineering  
December 1980

A

Approved for public release; distribution unlimited.

### Acknowledgements

An investigation of this nature required the assistance of many people. I wish to express my graditude to those who have so willingly provided advise and lent a willing hand. I wish to thank Major Alan A. Ross, my thesis advisor, for his perceptive comments and direction. The other members of my committee, Dr. Thomas C. Hartrum and Dr. Gary B. Lamont, provided exceptional support in offering assistance and in reading the report. Their efforts provided the necessary quidance that helped direct this investigation to its conclusion.

I am indepted to the AFIT laboratory staff, led by Mr. Robert Durham, and the efforts of Mr. Dan Zambon, who accomplished the rewiring of the MIME. Their support was exceptional and helped make it possible to complete this investigation on time.

Special mention also needs to be given Rick Purvis. As one of the original creators of MIME, his comments and suggestions, were particularly useful. His assistance is gratefully acknowledged, for without it, this project would not have progressed as far or have been as worthwhile.

The support, typing, proofreading, and assistance of my wife, Susan, were very instrumental in any success I may have received. I thank her as well as the rest of my family for their loving support throughout this investigation.

## Contents

	<u>Page</u>
Acknowledgements . . . . .	ii
List of Tables . . . . .	v
List of Figures. . . . .	vi
Abstract . . . . .	vii
I. Introduction. . . . .	1
Background. . . . .	2
Statement of the Problem. . . . .	5
Approach. . . . .	6
Organization. . . . .	8
II. Requirements Definitions. . . . .	10
Educationally Oriented Design . . . . .	10
Educationally Oriented Human Interface. . . . .	11
Operational Flexibility . . . . .	12
Writable Control Store. . . . .	14
Control Store Manager . . . . .	15
Controller. . . . .	15
Software Tools. . . . .	16
MIME Wirelist Formatter . . . . .	16
Executive Program . . . . .	17
MIME Modest Monitor (MIME/MM) . . . . .	17
Summary . . . . .	18
III. Hardware Realization. . . . .	19
Writable Control Store. . . . .	20
Control Store Manager . . . . .	21
MIME Documentation. . . . .	27
Summary . . . . .	28
IV. System Controller . . . . .	29
Introduction. . . . .	29
Controller Selection. . . . .	29
LSI-11 Microcomputer . . . . .	30
System Configuration. . . . .	31
LSI-11 Operating System . . . . .	34
Summary . . . . .	34
V. Software Tools. . . . .	35

Introduction. . . . .	35
MIME Wirelist Formatter . . . . .	35
The LSI-11 Executive. . . . .	37
MIME/MM Modest Monitor. . . . .	40
Summary . . . . .	42
VI. Results, Recommendations, and Conclusions . . . . . 44	
Results . . . . .	44
Recommendations . . . . .	48
Conclusions . . . . .	51
Bibliography . . . . .	53
Vita . . . . .	54

List of Tables

<u>Table</u>		<u>Page</u>
1	Control Store Modules . . . . .	22
2	Control Store Configurations . . . . .	22

List of Figures

<u>Figure</u>		<u>Page</u>
1	MIME Configuration (Phase II) . . . . .	4
2	Control Store Loading Process (Phase II) . . . . .	13
3	Present MIME Configuration. . . . . . . . .	26
4	Beginning LSI-11 System Configuration . . . . .	32
5	LSI-11 System Configuration . . . . . . . . .	33
6	LSI-11 Executive Program. . . . . . . . .	39
7	Control Store Loading Process . . . . . . . . .	41

Abstract

This report covers the continuing efforts to develop the Microprogrammable Minicomputer Emulator (MIME) into a more flexible and user oriented machine. This consisted of the development of a Microprogram Development System for the MIME as well as hardware enhancements. The amount of writable control store was increased and the ability to load and dump control under program control was added. An LSI-11 microcomputer was added to the system to allow the user to communicate with the other computers in the system, and to transfer files from one computer to another. Several software tools were also produced: (1) a program to semi-automate the generation of the punched cards used to document the MIME's hardware configuration, (2) an executive that runs on the LSI-11 to communicate with the other computers and the user, and to transfer the microcode files, and (3) updating the MIME Modest Monitor to communicate on a higher level and add several new features.

The results provide the user with a very powerful tool in the generation of microprograms. The operational flexibility of the MIME was increased and gives the user many new options in generating microprograms that will allow him to study microprogramming, emulation, and computer control.

MICROCODE DEVELOPMENT SYSTEM FOR THE  
MICROPROGRAMMABLE MINICOMPUTER EMULATOR (MIME)

I. Introduction

The Microprogrammable Minicomputer Emulator (MIME) was designed, constructed, and expanded as part of two separate Air Force Institute of Technology thesis projects (Ref 1 and 2). The MIME was designed as an educational tool that was specifically tailored as a user-oriented machine to aid the user in gaining experience in microprogramming. It was envisioned as a minicomputer that could be used as an aid to laboratory investigations of computer control and microprogramming (Ref 1: 5).

As the system developed, several important features were added to allow easier user access and use of the system. Further research and investigation was recommended following the conclusion of Phase II (Ref 2: 71). They recommended further enhancements and changes in two main areas. The first was the enlargement of the random access memory (RAM) in the control store area. The area available was only sufficient for small programs and for testing small routines. The second problem was that the translator, that was used to generate the microcode for the control store, had to run on a separate computer system. The translator

produced a file which contained the generated microcode. However, there was no path from the computer that ran the translator and the MIME. This prevented the microcode file from being transferred directly to the MIME. Even if the path had existed the MIME did not have the ability to load the RAM portion of the control store while in the run mode.

Thus the goals and aspirations of this investigation were geared to solve these two main problem areas. The ability to read from or write into control store memory while in the run mode was added, as well as increasing the amount of control store memory available. A computer based controller was added that allowed the microcode to be developed, translated, and loaded into the RAM portion of control store.

In order not to rewrite the information that is contained in the previous two investigations, the author is making the assumption that the reader is familiar with the concepts and definitions that are presented in the past investigations.

#### Background

The MIME was designed and built by Purvis and Yoho as part of their investigation (Ref 1). They were able to construct the MIME and debug the hardware to a degree that allowed them to test small microprograms and to determine that the MIME was capable of fulfilling their design goals. They also produced a design of how the MIME would be able to emulate another instruction set.

The second investigation by Hoyt and Myers (Ref 2) corrected most of the known errors in the MIME's hardware. They successfully emulated the MIL-STD-1750 instruction set which also required that the hardware be enhanced. In addition they produced some software tools, such as a translator for the MIME, that assisted the user in producing microcode quicker and with less errors. The architecture of the MIME following the conclusion of Phase II is shown in Figure 1.

During these investigations several shortcomings and possible future additions were identified. Some of these were that the MIME had no mass storage capability which could allow microcode to be stored temporarily, as well as only having a small amount of writeable control store in which to develop test code. In order to use this area the monitor had to be switched out such that the monitor became inoperative. The RAM then had to be loaded a word at a time while in the halt mode from a pad located on the front panel. In order to generate a microcode file, the translator was used to produce the file from the MIME language source code. This program is resident on the Control Data Corporation's Cyber Series Computer System (CDC) and produces several files, one of which is a microcode file. The user then had the option of burning this code in the EPROMs or loading it into the RAM via the front panel. The operation of producing a working microprogram thus became a tedious and slow project that was

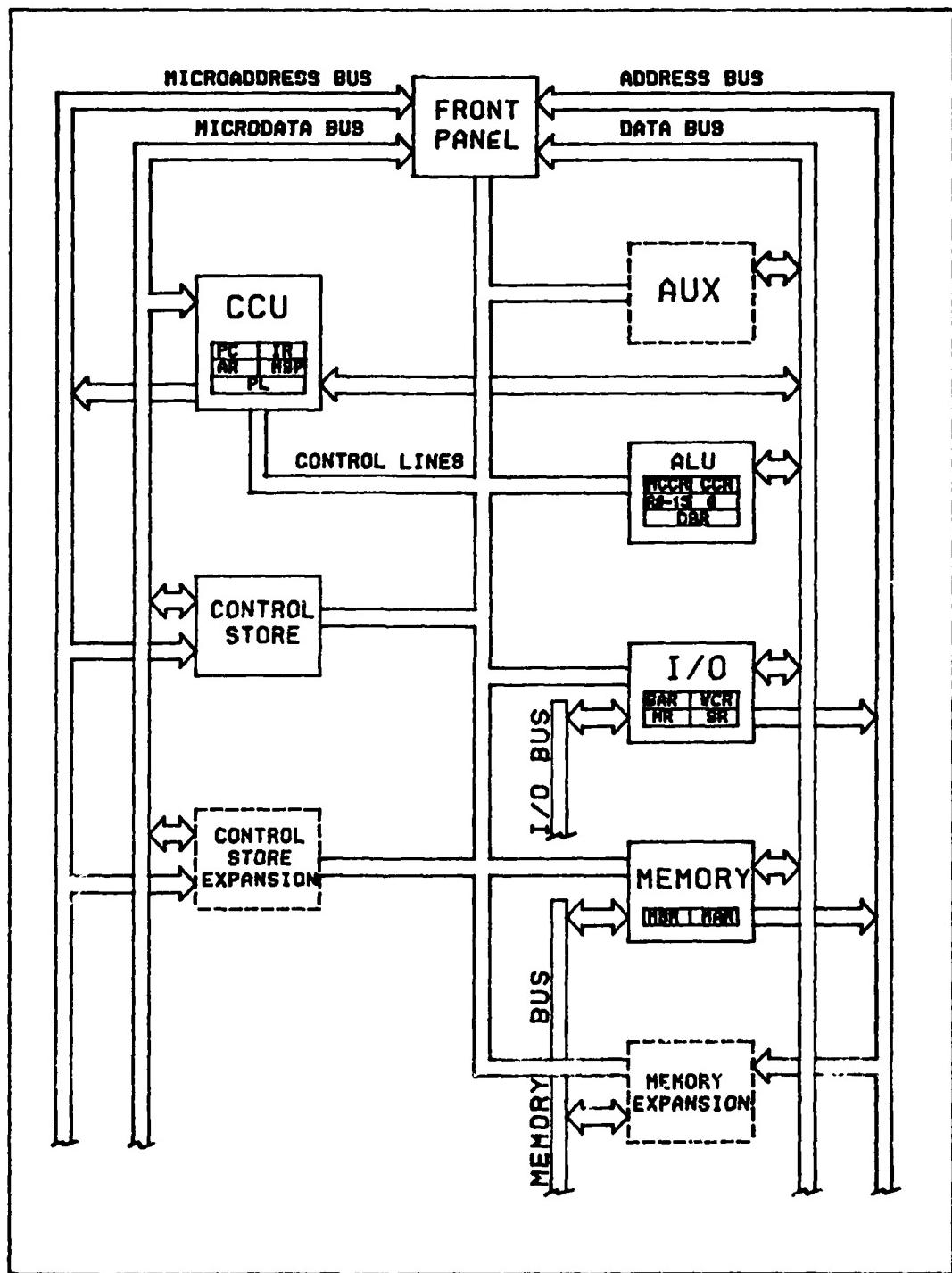


Figure 1. MIME Configuration (Phase II)

seriously hindered by user error and the time required to enter, verify, and run a microprogram. This then tended to reduce the educational value of the MIME since the process became unwieldy to produce and modify code.

The goals of this investigation were centered around the concepts of making the MIME more user oriented and restoring the MIME's educational value by making it easier to load, modify, and save microcode.

#### Statement of the Problem

The objectives of this investigation were to add the following capabilities to the MIME:

1. Capability to load and dump the control store memory (RAM) as well as to dump the ROM portion of the control store while in the run mode.
2. Add additional writable control store memory (RAM) to allow larger microprograms to be loaded and executed.
3. Add a computer based controller with mass storage capabilities in order to allow microcode to be "saved". The controller, a mini-computer, would then become a smart terminal that would interface with the MIME through the serial I/O port to transfer the microcode from mass storage to the writable control store. It would also transfer the microcode files from the CDC and store the file until it

was needed.

4. Develop and improve the MIME/MM (MIME Modest Monitor) into a more user-oriented system.

#### Approach

The following steps were taken in order to attain the goals of this investigation:

- MIME familiarization
- Requirements definition
- Hardware design and fabrication
- Controller integrated into system
- Support software generation

In order to determine the feasibility of this project, the architecture, operation, and hardware had to be studied. This was important so that the main areas of design and operation would become apparent. This entailed becoming familiar with the facets of the previous two thesis projects.

The next phase consisted of defining the hardware and software requirements. These requirements were then compared with the current MIME hardware configuration and the resources that were available to the user.

The pivotal point of this investigation was the addition of the ability to load MIME's control store while the MIME was in the run mode. The other goals could not be

met until this feature was added because they depended upon this ability. Thus the first phase consisted of adding the ability to load the RAM. After this feature was installed, a larger writable control store memory was added which gave the MIME the capability of having much larger microprograms loaded into control store.

Once the MIME had the ability to load its own control store memory, a computer based controller was added to the system. The LSI-11 manufactured by the Digital Equipment Corporation (DEC) was chosen as the controller. The controller provided the user with the ability to communicate with the CDC as well as the MIME. It allowed microcode files to be transferred from the CDC and stored on a disk. This file could then be transferred to the MIME and tested for correct operation.

The last phase consisted of modifying the Mime Modest Monitor (MIME/MM). Since the basic microcode development system was available for use, the enhancement of the monitor was used as a test case. The source code was modified, translated, and transferred to the LSI-11. It was then loaded in the MIME and tested. The code was modified several times until the desired results were obtained. This proved the overall system operation while providing the author with a positive experience. The monitor was modified to provide better communication with the user and allowed for better debugging of microprograms.

This investigation provided a learning experience with

three areas of computer design. It provided a hardware phase, a software phase, and provided the opportunity to interface several different types of computers.

### Organization

Chapter I provides the background and general goals for this investigation. It also gives a basic overview of the problem areas and the approach taken. Chapter II develops the requirements and discusses how they were derived. In Chapter III the implementation of the MIME hardware changes are presented along with some of the alternative solutions that were investigated. The selection and system configuration of the controller is found in Chapter IV. Chapter V discusses the software tools and programs that were required in order to complete this investigation. The results, conclusions, and recommendations for further investigation are contained in Chapter VI.

A major effort of this investigation was to update the appendices of Volume II and Volume III of the Hoyt and Myer investigation (Ref 2). The following appendices of Volume II and Volume III (Ref 2) were replaced by new updated versions:

- Appendix A, MIME User's Manual
- Appendix D, MIME Modest Monitor (MIME/MM) User's Manual
- Appendix H, MIME Monitor Source Code
- Appendix I, MIME Schematic Diagrams

- Appendix J, MIME Parts List
- Appendix K, MIME Wire-run Lists

The following appendices were added to Volumes II and III (Ref 2):

- Appendix D1, Controller's Executive User's Manual
- Appendix D2, MIME Wirelist Formatter User's Manual
- Appendix O, MIME Wirelist Formatter Source Code
- Appendix P, Controller's Executive Source Code

Volumes II and III are available from the Electrical Engineering Department, Air Force Institute of Technology (AFIT/EN), Wright-Patterson AFB, Ohio, 45433.

## II. Requirements Definitions

This chapter discusses the methods used to determine the requirements necessary to implement the goals stated in Chapter 1. In order to set the proper requirements, it was important to understand the goals and requirements that were developed to design the MIME architecture. This was essential so that the new requirements would be consistent with those already established.

The following discussion compares some of the established goals and how they compared with the configuration of the MIME at the completion of Phase II.

### Educationally Oriented Design

The designers stated that the top level MIME requirement was that it must be an educationally oriented, user-microprogrammable, small, general purpose digital computer (Ref 1: 9). Thus the architecture was designed and built around the concepts of close user interaction in order to learn the techniques of microprogramming and computer control.

The overall design and conception of the MIME's architecture and the detailed implementation was well conceived. The designers were very creative in their design of the architecture of the MIME, and allowed for future growth of the system. The MIME allowed the novice user to

start the learning process without too much background on how the MIME functioned internally. However, several areas of the design needed improvement.

#### Educationally Oriented Human Interface

The interface between the user and the MIME was one of the weaker areas. The interface was deisgned to be straightforward so that the user was not frustrated by the effort required to use the MIME. Convenience and the time required for data entry or retrieval was considered very important (Ref 1: 12). Tremendous forward strides were made with the addition of the MIME language and translator, and the MIME Modest Monitor by Hoyt and Meyer (Ref 2). These enabled the user to generate microcode much easier and also to verify correct results while the program was executing.

However, severe restrictions were still placed on the user in the area of loading the microprograms into the control store memory. The process to produce a microprogram required several different procedures depending on whether the program was to be loaded in the ROM or the RAM portion of control store memory.

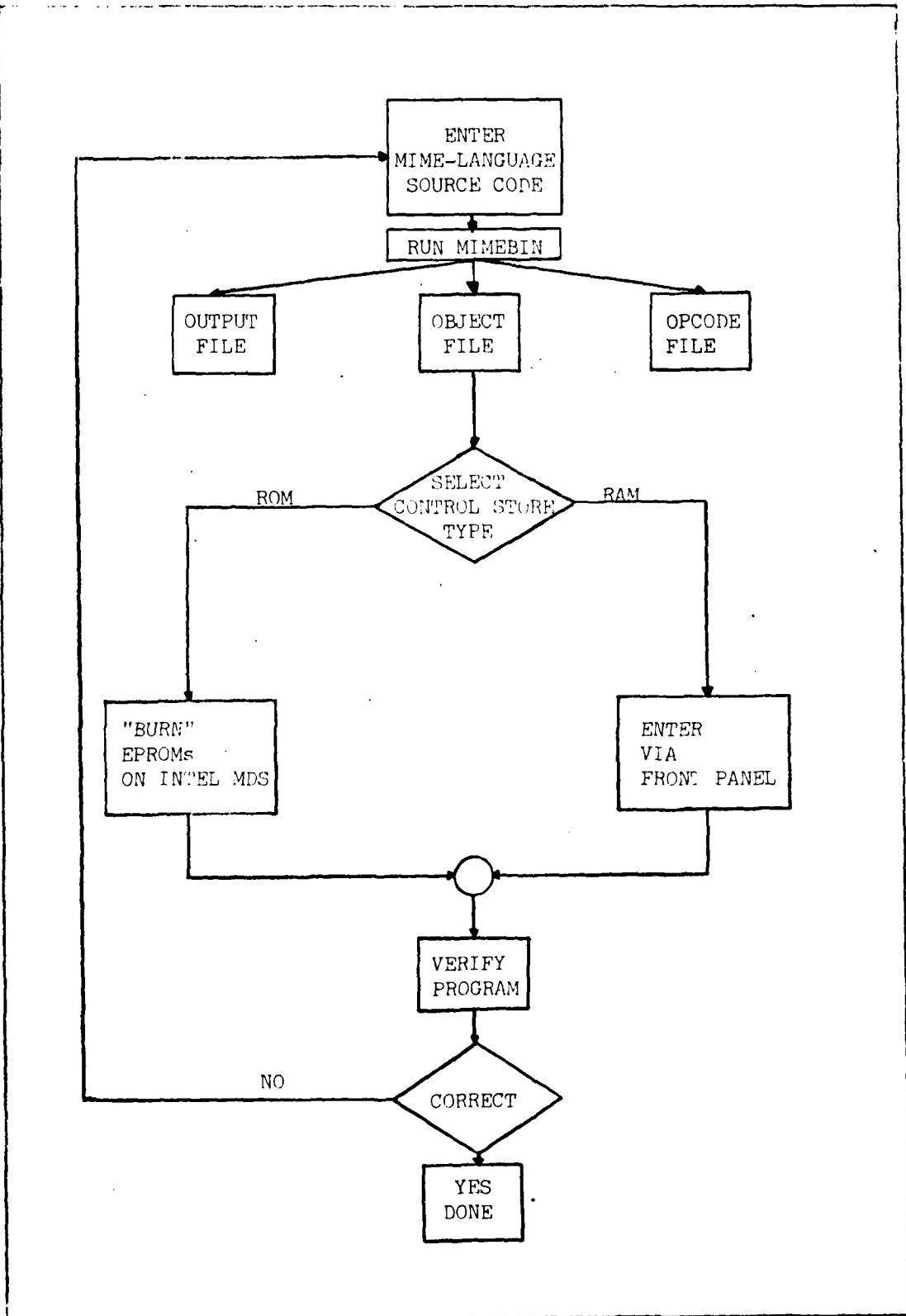
To load a microprogram in ROM required that the program first be written in MIME language source code. This code was then translated into microcode by the MIME translator program that ran on the Control Data Corporation's Cyber Series Computer System (CDC). The resultant microcode output was stored on the CDC. This file was then transferred

from the CDC and "burned" into the ROM chips via the Intel MDS (Microcomputer Development System) located at the Aeronautical Systems Division, Building 485. Finally the ROM chips were reinserted into the control store module and the program tested. If an error occurred, the complete process had to be repeated. This process is portrayed in Figure 2.

The process to generate microcode for the RAM was basically the same as for the ROM. The microcode could be generated using the translator program or the code could be generated by hand. The resultant microcode file produced by the translator was then printed. The user then inserted this code into the RAM using a pad located on the front panel. In order to use the RAM, the monitor had to be switched out of memory and the machine halted. If an error was made in the coding or insertion, the code had to be modified by hand and depending on the severity of the error, the entire contents of the control store might have been required to be modified. This process was not only a very tedious one, but also very error prone and time consuming. The very nature of the process tended to favor very small microprograms.

#### Operational Flexibility

The requirement of operational flexibility was also levied against the MIME design. This requirement stressed



**Figure 2. Control Store Loading Process (Phase II)**

the ability of the user to accomplish his purpose with minimal laboratory setup or initialization (Ref 1: 12). Not only should this be true, but the manner in which the user controls the operation of the system must also be flexible.

There were several areas in which the operating system tended to frustrate the user. If the user desired to run a microprogram using the RAM portion of the control store, he had to disable the MIME Modest Monitor. This was due to the overlapping of the addresses of the RAM and the EPROM that contained the monitor. This meant that the debugging tools supplied by the monitor were not available under this mode of operation. The user had no way to inspect many of the internal registers. This caused many hardships when a program was being debugged.

Since there were only 256 words of RAM, the user was limited to small microprograms. The manual entry method also tended to restrict the size of the microprogram. The only way to produce a large program was to load it into EPROM and test it. These limitations placed restrictions on how the user developed microprograms, thus reducing the operational flexibility of the system.

With these deficiencies in mind, the requirements of this investigation were developed.

#### Writable Control Store

In order to allow larger microprograms to be loaded in real time, increasing the amount of writable control store

(RAM) was set as a requirement. This would allow more complex and expanded programs to be loaded and tested. The contents could be rapidly altered, since the memory could be changed by reloading it either from the front panel or by using an alternative method. This approach would give the user several options on how the program was coded, loaded, and tested.

#### Control Store Manager

To overcome the tedious manner in which a program was entered into the RAM portion of control store, the requirement was established for a manager that would read from or write into control store while in the run mode. This provides the MIME with the ability to display the contents of control store or allow the contents to be altered under program control. This allows programs to be uploaded or downloaded by computer or user control using the external I/O port instead of the front panel.

#### Controller

The requirement for a computer based controller was established to allow the user to more effectively use the available tools. He needed to communicate with the CDC as well as the MIME. He must also have the capability to transfer the microcode files from one computer to another. But before this requirement could be realized, the ability

to load or dump control store must first be incorporated. The requirement for the controller specifies that a small computer system should be used as a smart terminal. It must have the ability to communicate with the user, the CDC, and the MIME. The controller must have access to mass storage which gives the user the ability to store microprograms until the user requires them. It must also have the capability to transfer files from the CDC to itself, and then transfer the files to the MIME.

#### Software Tools

Several software tools were required to increase the operational flexibility and the ease of user interaction. In order to realize several of the hardware requirements, some additional software programs were required. The paramount thought was to produce the tools that would save time and reduce the effort required to produce and test microprograms.

#### MIME Wirelist Formatter

In order to rewire the MIME circuit cards and also produce the documentation for the MIME, the changes were entered a card at a time into the documentation data base. Each card required a great deal of human interaction in order to determine chip and pin number as well as board location for that pin. It was felt that the computer would

be much better at this monotonous and slow job. Therefore, a requirement was set for writing a program that would allow the chips to be defined and then from minimumm inputs, determine the required output format and print it.

#### Executive Program

To run the controller, a program had to be designed that would allow it to communicate with the MIME. Thus, the requirement developed to write a program that would talk to the MIME as well as upload or download microcode. It also had to have the ability to communicate with the CDC in order to obtain the microcode file generated by the MIME language translator.

#### MIME Modest Monitor (MIME/MM)

The MIME/MM was designed as a minimum-capability interactive monitor. The authors defined minimum-capability as the fewest number of essential capabilities required in a monitor, but that still provided the user with a useful tool. The monitor was provided to allow the user to follow the progress of a microprogram's execution and to allow register and memory manipulations from a user terminal (Ref 2: 51-53).

The designers, with the idea of keeping the size of the monitor to a minimum, had the MIME/MM communicate with the user in a shorthand form. All errors were treated in the

same manner. The resulting error message gave the user no hint at what had caused or created the error condition.

The above design decisions left the MIME Modest Monitor with several shortcomings. It communicated on a low level, in a very ambiguous manner, with the user. Since the address of the monitor overlapped the address of the RAM, the monitor had to be disabled when the RAM was used. This set the stage for the requirement to reorganize the addressing of the control store to allow the monitor to remain present when RAM was being used and to increase the communication level with which it interacted with the user.

#### Summary

This chapter has presented a discussion on how the requirements were identified for this investigation. Chapter III presents a discussion of the realization of the hardware additions.

### III. Hardware Realization

This chapter presents a brief discussion of the alternatives studied and the solutions chosen to fulfill the hardware requirements stated in Chapter II. The distinction between hardware and software changes to implement a specific function is not as clear in a microprogrammable as that present in a nonmicroprogrammable machine. By merely changing the pattern of bits and the order in which they arrive, a microprogrammable computer's function and operation can be changed. Changes are also possible by adding additional hardware and then using or redefining the microcode bits to operate the added hardware. Thus, a microprogrammable computer by its very nature lends itself to modification and changes. This proved to be very useful as the requirements were analyzed for possible solutions.

#### Writable Control Store

As was stated in Chapter 2, the MIME only had 256 words of RAM (writable) control store. In order to use it, the monitor had to be disabled since the addresses of these two areas overlapped. The ROM that contained the monitor was located at address 000-3FF (HEX). The RAM was also located at address 000-0FF (HEX) which meant that only one could be used at a time. When the MIME executed the RAM portion of

control store, it merely ran on into the ROM addresses that exceeded OFF (HEX).

The control store memory was contained on two internal modules in the MIME. These two modules were labeled as CS1 and CS2. The CS1 module contained 1K of EPROM and 256 words of RAM control store. The CS2 module contained 3K of EPROM control store, thus using CS1 and CS2 the maximum amount of control store was available (Ref 2: A23-A26).

The implementation was straightforward for the addition of more control store memory. The Intel 256 X 4 bit Static RAMs (2112) were replaced by Intel 1024 X 4 bit Static RAMs (2114A). This gave the CS1 a total of 1K EPROM and 1K of RAM.

The EPROM of the CS1 Module was used as the first K of control store, address 000-3FF (HEX). The decision was made that the monitor would never be over 1K words long and would reside in the first K of control store. Several factors guided this decision. The MIME had been designed to run with only a minimum number of modules installed. The CS1 Module was required to be installed while the CS2 Module was optional. This meant that if the monitor was to be available at all times then it must be loaded in EPROM on the CS1 Module. Since it was desired that RAM also be available on a constant basis, some RAM needed to reside on the CS1 Module as well. But due to the physical size restriction of the module, it could only contain 1 K of each type. This limited the size of the monitor to 1 K words.

The RAM on the CS1 Module was addressed at 400-7FF (HEX). Since this overlapped the second K words on the CS2 module, a toggle switch was provided to select the desired type of control store (EPROM or RAM). An additional 1K word of RAM was added to the control store circuit card (CS2B) which replaces the CS2 when the user wishes to write into control store or to dump CS under program control. Tables 1 and 2 summarize the available control store memory and the possible variations. Additional information can also be found in Appendix A (Ref 2).

#### Control Store Manager

Before any hardware changes could be made, the architecture, control signals available, data paths, and MIMIC operation each had to be studied in detail so that the modifications would be compatible to the existing machine.

As stated in Chapter II a control store manager was needed that would allow the control store to be read from or written into while in the run mode. This manager, however, had to be able to communicate with some outside source such as another computer or a user. The only external path for the data flow was the single serial I/O port. Therefore, the manager required a data path from itself and the data bus, which interfaced with the I/O module. The manager was also required to interface with the Microaddress Bus and the Microdata Bus. This meant that the design had to be placed

MODULE	TYPE OF CONTROL STORE
CS1	1 K EPROM 1 K RAM
CS2	3 K EPROM
CS2B	1 K RAM

Table 1. Control Store Modules

ADDRESS	CS1		CS1/CS2		CS1/CS2B		
	F	*	R	F	R	F	R
000-3FF	EPROM	EPROM	EPROM	EPROM	EPROM	EPROM	EPROM
400-7FF		RAM		EPROM	RAM		RAM
800-BFF				EPROM	EPROM		RAM
C00-FFF				EPROM	EPROM		

\* Refers to position of toggle switch located on CS1. (F) Forward (R) Rear

Table 2. Control Store Configurations

on a module that had access to all three busses.

Two possible solutions were studied to implement this requirement. The first considered was the addition of another microprocessor into the MIME. It would have used a ROM program to access the data bus and to cause the MIME to halt at the appropriate times to allow the MIME's control store to be read or altered. It would have required two buffers to store the incoming data. Since the data bus is only 16 bits wide, the controller would need four words to be sent to load the 64 bit control store word in the case of writing into Control Store. On the other hand, when the Control Store was read, it would have obtained 64 bits from the control store and then sent 16 bits at a time to the data bus where the MIME would take over and process the data. This approach would have used the entire circuit board in order to place the required processor, ROM memory, and buffers on the card. This restricted the MIME to whatever control store memory was present or could be added on the CS1 module.

The processor would have used the same control signals that MIME used to access the required areas. This meant putting the MIME in the wait mode, enabling the processor which would control the operation until the process was complete, and then return control to the MIME.

The second solution considered involved the use of random control logic. The process to upload or download control store is a straightforward process that requires

very simple control logic. This logic would be relatively simple to implement in random control logic, and with the use of some microprogrammable bits, pass the data from one module to another. This approach also required the use of buffers due to the mismatch of the size of the micro-buses and the data bus. It likewise required MIME to be placed in the wait mode while the control store area was being accessed.

Several advantages resulted from the second approach:

- The MIME would use its own microprocessor to manipulate the data and control the flow of information.
- The circuit would not have to be programmed to be used and would avert some possible timing problems between processors.
- The type of chips required to implement the logic were much smaller in size, number, and complexity, thus leaving some spare room on the card for additional control store memory on the module.
- The maintenance would be much simpler and much more straightforward. This was in keeping with the requirement that the MIME needed to be educationally oriented.

The main disadvantage of the second approach was that it may require a hardware change if the operation of the manager is altered. To change the operation of the

processor may only require changing the program in the ROM memory. This meant that it was less flexible than the first solution.

After both the alternatives had been studied, the second solution was chosen. It was felt to be more in line with the requirements and would also allow the size of control store to be expanded since there was room left over on the circuit card.

The control store manager was added to the architecture as shown in Figure 3. It provided the necessary data path between the data bus, the microaddress bus and the microdata bus.

The control store manager contains three buffers. The buffers were constructed using Octal D-Type flip flops 74LS374, one buffer to hold the 12 bit microaddress, one buffer to receive data from the data bus and pass it to the microdata bus, and the third to receive data from the microdata bus and pass it to the data bus.

The required hardware, for the random control logic, and its operation are provided in the MIME User's Manual (Appendix A, Ref 2). Each operation, load and dump control store, required separate logic since the process was somewhat different.

The control signals required to route the data from the buffers in the control store manager and the signals to control its operation were:

- load the microaddress buffer

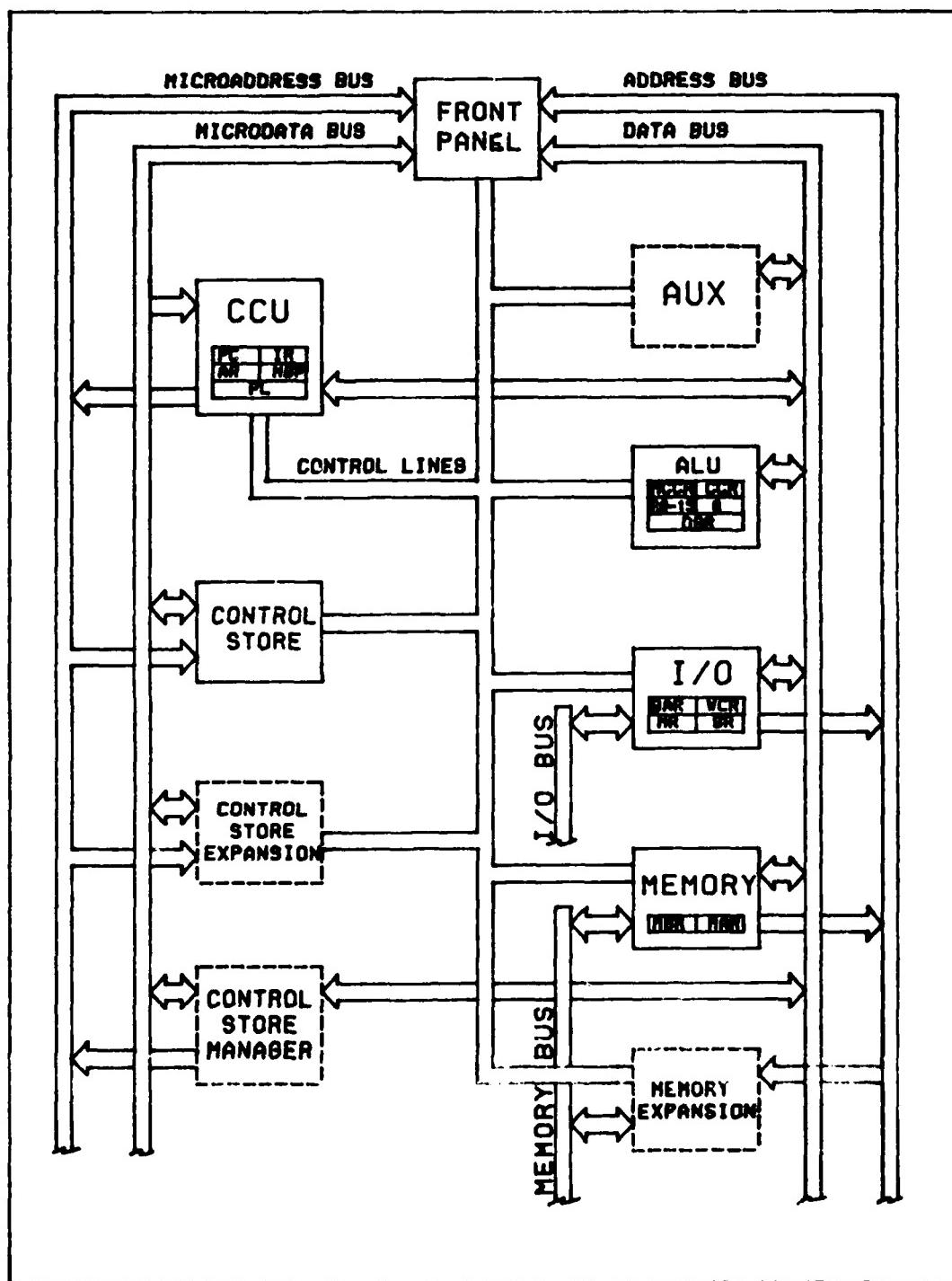


Figure 3. Present MIME Configuration

- load the buffer between the microdata bus and the data bus
- load the buffer between the data bus and the microdata bus
- start the load control store routine
- start the dump control store routine

These control signals were implemented by using 5 spare functions defined under the AUX Function block (Ref 2: Table BIII, B-7):

UDAFB  
UDAFC  
UDAFD  
UDAFE  
UDAFF

This allowed the MIME to control the operation of transferring the data from module to module and also load or dump control store. It also made use of the existing data paths and modules for bringing the information from the external world into the MIME.

The implementation required a small micropogram for each operation (load or dump control store) to be added to the MIME Modest Monitor. The details of these programs can be found in Chapter 5.

#### MIME Documentation

Due to the wiring changes, the MIME documentation had

to be updated. The schematics of the affected circuit cards had to be redrawn and an additional schematic for CS2B inserted. The MIME Wiring List was also updated and verified using the MIME Wire-List program. These new versions were used to replace the previous ones. The MIME parts list was modified to reflect the additions/deletions and the changes to the circuit cards. This insured that the documentation reflected the current configuration of the MIME.

#### Summary

This chapter has presented the development of the required hardware modifications that were needed to fulfill the stated requirements. The approaches taken were also presented. The selection and system configuration of the computer based controller is discussed in Chapter 4.

## IV. System Controller

### Introduction

This chapter discusses the selection and system configuration of the controller. As discussed in Chapter 2, a computer based controller was necessary to interface the user with the CDC and the MIME. It also needed the capability of accessing a mass storage device in order to store microcode files. In this capacity the controller becomes a smart terminal (able to store and process data independently) that allows the user to communicate with the CDC or the MIME. The controller thus becomes the heart of the system configuration and gives the user a greater degree of flexibility.

### Controller Selection

After considering the requirements that were stipulated for the controller, the following required capabilities for the controller were established:

- It had to have the ability to communicate with the MIME, and the CDC.
- It had to have the ability to communicate with the user (such as through a console device).
- It had to have the ability to print out

required status and information.

- Mass storage was a requirement in order to store the microcode files.
- The controller had to also be available on a fairly consistent basis in order that the software and the system design could be completed.
- It needed the capability to compile PASCAL source programs so that the software tools developed during the previous investigation and this investigation could be added to the system at some future date.

The actual selection of the controller was fairly simple since the only available minicomputer that could meet the above stipulations was the Digital Equipment Corporation (DEC) LSI-11 microcomputer. Because it was also highly recommended by the thesis advisor and Digital Engineering lab director, the LSI-11 was an automatic choice.

#### LSI-11 Microcomputer

The LSI-11 is a 16-bit microcomputer manufactured by Digital Equipment Corporation (DEC). The LSI-11 system components include separate modules that can be added individually to configure the LSI-11 to the required application (Ref 3). This allows the system designer to build the computer system around his specific requirements. A required feature can be added or deleted by merely adding

or deleting a printed circuit module.

To fulfill the requirements that were stipulated for the controller, the following modules were installed:

- Microcomputer Module
- 28K RAM Memory
- 4 Port Serial I/O Module
- Floppy I/O Module

#### System Configuration

The LSI-11 configuration had several shortcomings at the beginning of the investigation. While all the required modules were available, only part of the capabilities had been utilized. The main shortcoming was in the I/O area; the 4 port serial I/O module had only one port in use. The printer was daisy chained from the console which meant that whatever was viewed on the screen was also printed out. Figure 4 represents the system configuration at the beginning of this investigation.

The main reason for the system configuration was the lack of an operational printer I/O handler. In order to connect the printer to a separate channel, an operational handler had to be provided. A partially developed printer I/O handler had been developed as part of a lab project by Croasdale and Staubs (Ref 4). This handler was corrected, verified, and incorporated into the operating system.

The remaining two serial I/O ports were assigned to the CDC and to the MIME as shown in Figure 5, and an I/O driver

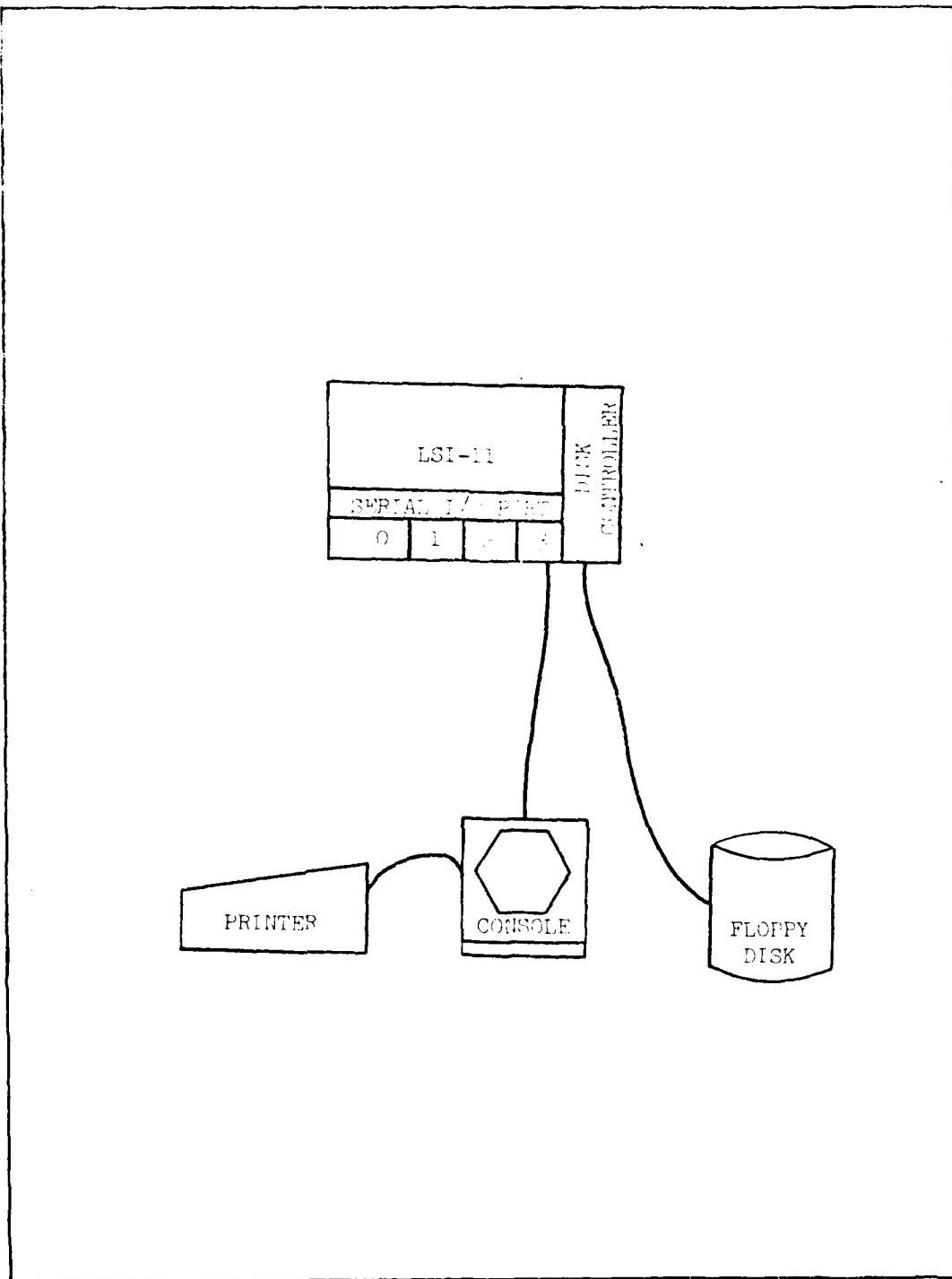


Figure 4. Beginning LSI-11 System Configuration

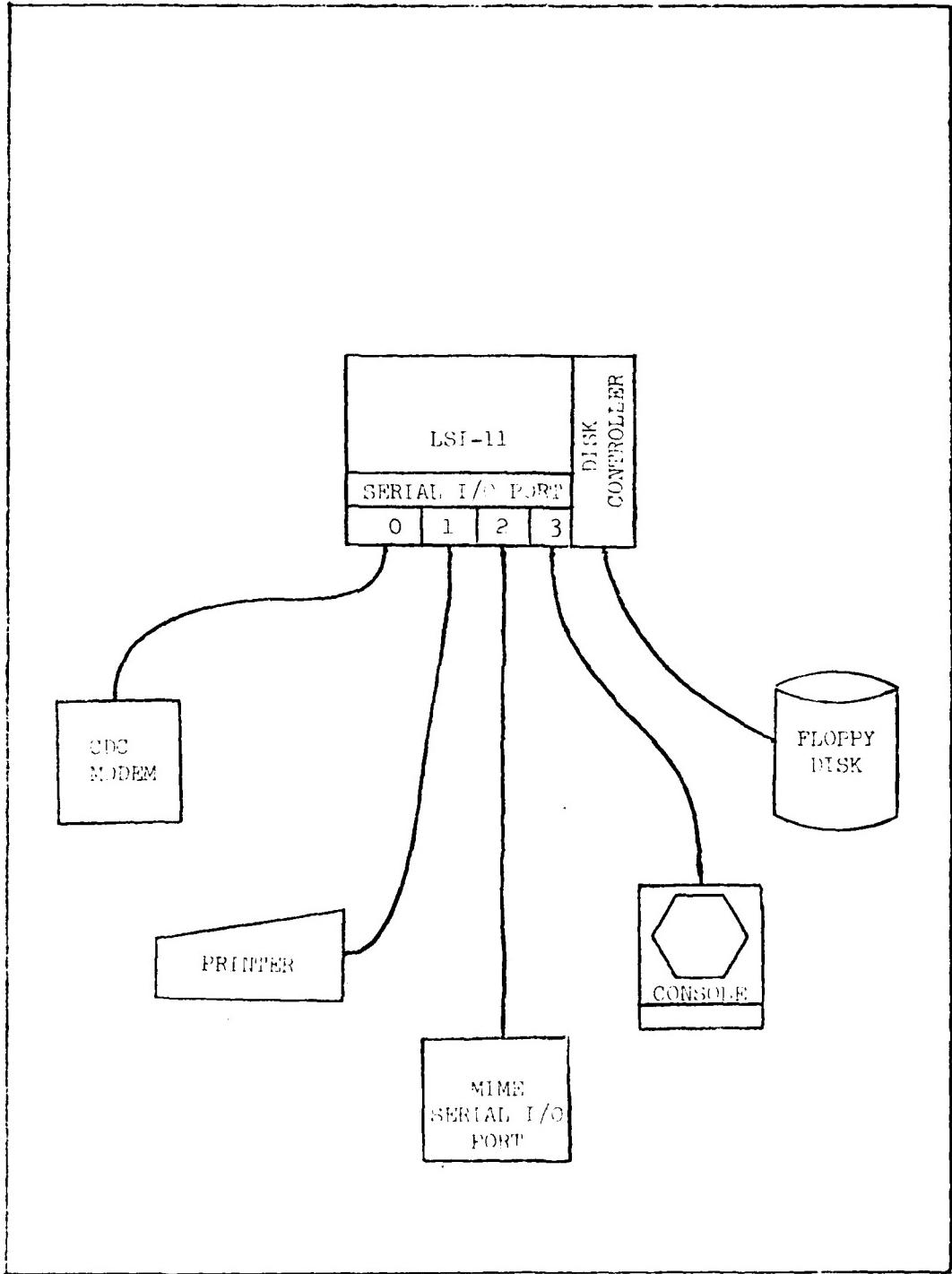


Figure 5. LSI-11 System Configuration

routine was produced as part of the executive that is resident on the LSI-11. This program is discussed further in Chapter 5.

#### LSI-11 Operating System

There are two operating systems available to the LSI-11. The standard DEC RT-11 version 3 and the UCSD PASCAL (Ref 5).

A printer I/O handler was incorporated on both operating systems. The user can now use the LSI-11 to run either assembly programs using the RT-11 or PASCAL programs using the UCSD PASCAL operating system. The PASCAL operating system is very important because it gives the LSI-11 the capability of having the software tools that have been procuded during the different investigations, added to the system. This will allow the user to generate the microcode from a single point.

#### Summary

This chapter has presented the development of the required capabilities, the selection and system configuration of the controller. It discussed the capabilities of the LSI-11 and the beginning and ending system configurations. Chapter 5 covers the software tools that were developed.

## V. Software Tools

### Introduction

The development and design of the software tools produced during this investigation are described in this chapter. The three software tools produced were (1) a program to semi-automate the generation of the formatted wirelist as well as the punched cards used as a documentation resource; (2) an executive which runs on the LSI-11 that interfaces the user with either the CDC or the MIME; and (3) an improved version of the MIME/MM Modest Monitor that allows control store to be read into or dumped from and is more user-oriented. These tools were designed to add greater flexibility to the overall system and to assist the user in generating microcode.

### MIME Wirelist Formatter

The first tool produced was a by-product of the hardware modification phase. The hardware changes were incorporated on two internal circuit modules within the MIME. One of the modules was completely rewired, while a new module was created for the control store manager and some of the additional writable control store. These additions and changes were documented in the wirelist that was part of the Phase II investigation. This wirelist has a

two-fold purpose: (1) it is used as a guide during the wiring phase and (2) it serves as the documentation for the MIME's configuration.

The process to produce the wirelist was slow and subject to error. Each wire was translated into two punched cards, one describing where the wire originated and its destination, and the second card as a mirror image of the first. This served as a consistency check and helped verify the correctness of the wirelist. The information that was needed for the card and the actual punching were generated manually. This took a great amount of effort and produced many errors.

After attempting to generate the documentation for the required wiring changes and finally winning the battle on the first hardware module (the wiring was finally correct as well as the documentation), it was decided that there had to be a better way. It was felt that if the required data was entered in a very simplified form, that the computer would be able to generate the required information, format it, and then punch out the cards much easier as well as quicker than the manual approach. This approach would also produce a much more correct product as long as the input data was correct.

The program was written in PASCAL using a top down design approach. PASCAL was chosen as the source language because it is very structured and lends itself to a structured approach to the problem. Future installation on

the LSI-11 was also a chief consideration. PASCAL insured easier software maintenance due to the structured design and should make it simpler for the user to understand the design as well as the operation. The program was installed on the CDC since the output it generates is used as the input for the MIME program which produces the wirelist documentation. The problem was basically data manipulation. It required that the chip parameters first be defined and then a wire read and the output information generated. This resulted in the input data being broken down into two separate types: (1) the first section identifies what module is being defined and the chip parameters that are required to define the size and location of the chip; and (2) the second section listed the wire changes or additions. The data was then processed, formatted, and placed in the output file where it could either be printed or punched, depending on the requirement. Further details of this program can be found in Appendix D2 (Ref 2).

#### The LSI-11 Executive

In order for the user to have access to the files on the CDC as well as to communicate with the MIME, the LSI-11 Microcomputer was placed in the system to act as a smart terminal. It allows the user to regulate with whom he desires to communicate as well as the desired action that is required. To use the LSI-11, an excutive was required.

While the LSI-11 already had an operating system, the operating system did not have the ability to communicate with the CDC or the MIME. Thus, the requirement was established for an executive that would allow the LSI-11 to communicate with the other two serial I/O channels which could be connected to the CDC and to the MIME. The executive was also required to perform tasks such as transferring and storing the microcode file from the CDC. This file was then required to be sent to the MIME. These requirements were used to produce a structure design.

Due to the nature of the design and the requirement to communicate with the serial I/O channels, the design was implemented using the RT-11 operating system and written using assembly language. While this language does not lend itself to a structured implementation, the code was modularized as much as possible. Assembly language was chosen because it is a very straightforward approach to handle the large amount of I/O processing for this program. The basic assembly language for the LSI-11 is very flexible and allows very complex I/O operations while using simple procedures. This, coupled with lack of documentation for the PASCAL operating system, became the basis for the choice for using assembly language.

The design was organized in basically the following main modules: (1) Communication Module and (2) the Application Programs Module. See Figure 6.

The design allowed the user to communicate with either

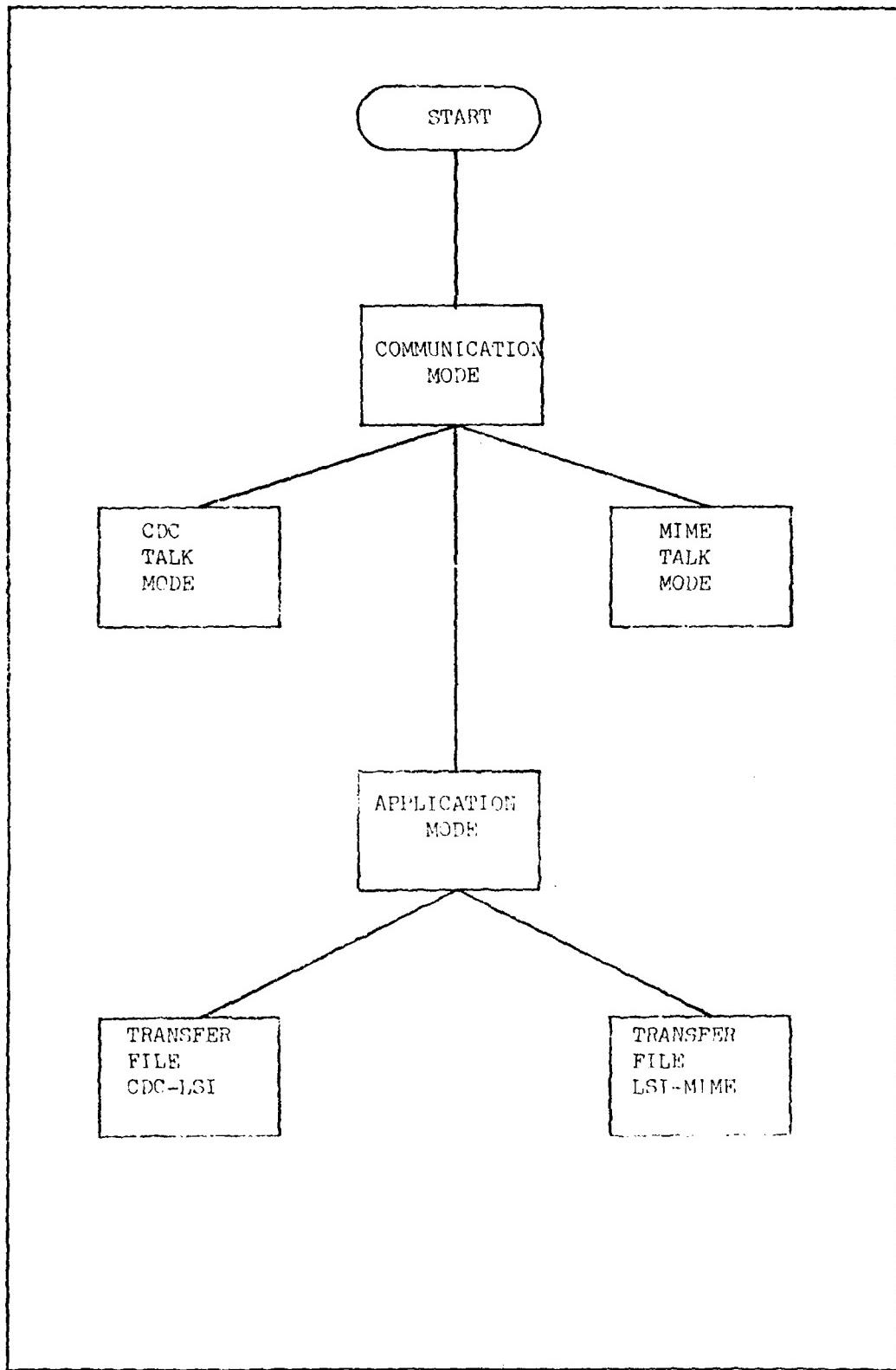


Figure 6. LSI-11 Executive Program

the CDC or the MIME, or to transfer a microcode file. The process of producing a microprogram now follows the process as depicted in Figure 7. Additional details of this program can be found in the Controllers Executive Users Manual (Appendix D1, Ref 2).

#### MIME/MM Modest Monitor

The last tool developed was a modification of the MIME/MM Modest Monitor. The MIME/MM had been designed to interact (communicate) with the user on a very low level. All interactions were in a shortened form which required more knowledge of the monitor that should be expected of a user. All errors produced the same error message which gave the user no information on how the error had been generated.

The monitor allowed dumping and loading registers, dumping and loading memory to a terminal for display or storage and exiting the monitor to the user program (Ref 2: 53). All of these functions were basically left intact.

The major emphasize of the modification was to upgrade the manner in which the monitor communicated with the user and to add two additional features that would allow the control store to be read or written into while in the run mode.

The communication level was upgraded primarily by expanding the monitor's response into a somewhat more meaningful form. The user now is not required to know as

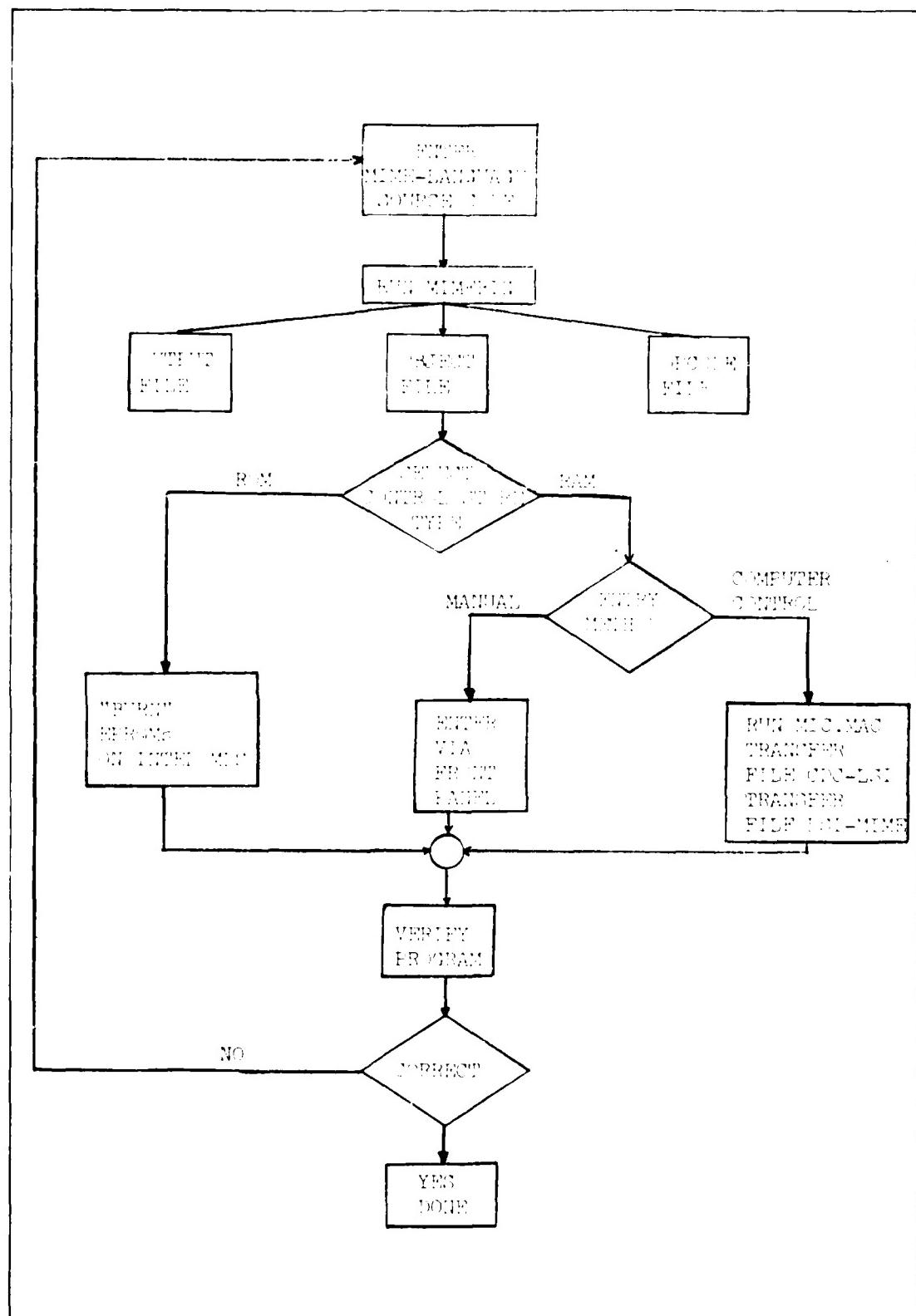


Figure 7. Control Store Loading Process

much about the monitor and what it expects as input.

The two additional features, Load Control Store (LC) and Dump Control Store (DC) were added. The hardware installed during the hardware phase required that its buffers be loaded at a set time and in the proper sequence. The control points are governed by the use of microinstructions that control the flow of information. These two features, LC and DC, allow the hardware added during the hardware phase to function properly. The LC and DC function by commanding the data to be read in or out, loaded or dumped into the buffers and then read from or written into the control store memory. These functions allow the user to display the contents of control store on a terminal, or another computer acting as a terminal, or to load the contents of control store while the machine is executing. Appendix D (Ref 2) contains the MIME/MM Users Manual and gives additional information on the use of this program.

With the incorporation of the load and dump control store features into the monitor, along with the added hardware, the user now has at his disposal a very powerful mechanism to generate, modify and execute microprograms.

#### Summary

This chapter has presented the design goals and rationale for the software tools developed for the MIME--the MIME Wirelist Formatter, the LSI-11 Executive, and the

modified MIME/MM Modest Monitor. These tools were designed to have general application to any microprogramming effort and to ease the task of generating, modifying, and producing microcode.

## VI. Results, Recommendations, and Conclusions

The objectives of this investigation were to provide the user with greater flexibility in designing, debugging, and executing MIME microprograms. To meet these requirements, a microcode development system was added to the resources available to the user of the MIME for the development of microprograms. The attainment of this major goal was met by the incorporations of the following subgoals:

- Adding the capability to load and dump control store memory under program control.
- Increasing the size of the writable control store memory.
- Introducing a computer based controller that allows the user to communicate with the CDC, the MIME, and transfers the microcode files.
- Modifying the MIME/MM (MIME Modest Monitor) in order to make it communicate on a higher level and adding the new features required to allow loading and dumping of control store.

This chapter discusses the results of this investigation, provides recommendations for further research, and presents the conclusions reached.

### Results

The investigation results are presented in the following paragraphs:

Control Store Manager- At the end of Phase II (REF B), the MIME did not have the ability to load or dump the contents of control store memory under program control. The key pivot point of this investigation was the addition of the control circuits that allowed this feature. All the following subgoals were dependent on the realization of the ability to load and dump the contents of the control store memory under program control. The control circuits were designed, debugged, and incorporated into the hardware of the MIME. The design proved to be correct and allowed this required ability. This now gives the user the ability to have the contents of control store transferred to an external device such as a terminal or serial I/O port of another computer. While the ability to download a program from the MIME to the LSI-11 does not yet exist, the user can display the contents of control store on the console. The contents of control store (RAM) can now be modified or loaded from a terminal or another computer. This relieves the user from the tedious and error-prone process of microcode insertion via the front panel.

Writable Control Store- The amount of writable control store at the beginning of this investigation was only 256 words. This tended to limit the user to small microprograms if he desired to use the RAM portion of the control store memory. He was also limited by the manner in which the code

was required to be entered. The size of the writable control store memory was expanded to 2K words to allow larger programs to be loaded. This expanded area can be quickly loaded using the LSI-11 or by using the console. More complex micropograms can now be loaded, tested, and modified in a much shorter period of time. This greatly increases the user's capability in the area of flexibility and microprogram generation.

Computer Based Controller- In order to allow the user to communicate with the CDC, which runs the program to translate the MIME language into microcode, and interface the user to the MIME, a computer based controller was added to the system. The LSI-11 was chosen as the microcomputer because it had the necessary capabilities in the I/O area, the operating systems (the basic RT-11 operating system, and UCSD PASCAL operating system), and access to a terminal and mass storage. The addition of this controller became the heart of the microcode development system. An executive program was installed on the LSI-11 that interfaced the user to either the CDC or to the MIME. He can now control the execution of the translator, monitor the results, and transfer the contents of the microcode file to the LSI-11 where it is stored on a disk. The user then can change the operating mode to communicate with the MIME/MM that is executing on the MIME. Following an initializing step, the microcode file can be transferred to the MIME and loaded in the writable area of control store. This can then be

executed, correct operation verified, and if necessary return to the monitor mode. If the program does not perform as expected, the microcode file can be modified using the LSI-11 editor, or regenerated using the translator on the CDC. The file can again be transferred and reloaded in the MIME and executed. This iterative cycle can be repeated until an operational program is produced. The time required to perform the above operation is greatly reduced from the previous manner. The user can now generate new microprograms much faster, with fewer errors, with a much simpler process.

MIME Modest Monitor (MIME/MM)- The modification of the monitor was required by the low level with which it communicated with the user. It communicated in a shortened form and treated all errors the same. This left the user with the requirement of knowing how the monitor was designed, and required him to understand what the monitor expected as input. The communication level was upgraded in order to relieve the user from these requirements. He is no longer required to understand what input is required since it is now displayed on the terminal. The error conditions are treated individually such that now the user is aware why the error condition occurred or what generated it. Also two new features were added to the monitor that allows the user to load and dump control store under program control. The addressing of the control store memory was reorganized such that the monitor can remain resident whether the RAM or ROM

portion of control store is being utilized. This provides the user with the same capabilities whether the RAM or the ROM portions are used to store the microcode and execute the program.

Documentation- A major portion of this investigation was to produce an updated version of the documentation. This included adding the changes that were required in the wirelist tables, updating the parts list, and adding the new features and programs that are now available. New schematics were drawn as well as updating the modified module's schematic. The MIME's User Manual was updated with the new changes incorporated, and two additional user's manuals were added that instructed the user on the use of the MIME Wirelist Formatter and the LSI-11 Executive.

As a passing note to the reader, the Microcode Development System was used to develop the modified version of the MIME/MM. It is felt that the system allowed this revision to be accomplished in a timely manner and proved to be a positive experience.

#### Recommendations

This section provides several recommendations for further investigations involving the MIME and its utilization.

While some hardware and software additions would increase the usefulness of the MIME, the author feels that the main recommendation would be to use the system and tools

already provided. The MIME should be used as a lab resource for a class on microprogramming. This would provide for some user comments on how well the system has been designed and also give some additional insights into what is really necessary in order to make the MIME a more useful tool. It would be a shame to let the MIME and the tools developed during the past investigations lay idly on the shelf and grow old due to lack of use, and would represent a tremendous waste of the human effort that produced the system and the state that it is now in. THUS-- PLEASE USE IT!

Following this period of use, the following hardware and software changes could be added as well as those identified as the system is exercised:

- Since the UCSD PASCAL operating system is available for use on the LSI-11, the different programs (i.e. the MIME language translator and the MIME Wirelist Formatter) could be added to the system. This would allow the user to run all the required software on the LSI-11 and would delete the requirement to run the translator on the CDC and then transfer the microcode file to the LSI-11.

- If the translator was removed from the CDC, then some additional path would have to be provided to allow the ROMs to be loaded from the LSI-11. This would then allow the user to load both types of memory from the same microcomputer and further reduce the time required to produce microprograms. The user could then generate, debug,

and test his microprograms using the RAM portion and then "burn" the completed microcode in the more permanent EPROM.

- Additional features could be added to the executive that runs on the LSI-11. The path to load EPROM could be added as well as the path to download programs that have already been loaded into control store. The ability to store the macro memory contents and to load new macro programs via computer control would be very useful. The Monitor already has the ability to load and store information to a tape device. It would be relatively simple to use the LSI-11 as the tape device and have it load or store the information on the disk.

- Test routines could be generated that test particular areas of the MIME. These self-diagnostic microprograms could be saved on disk and loaded into the writable control store via the LSI-11. This would give the user the ability to determine the "health" of the MIME. Each routine could be developed to test specific areas such as I/O, macro memory, ALU operation, control store memory, as well as the other MIME modules.

- The size of macro memory is only 1k words. This area could be increased in size up to 64K. This would allow larger macro programs to be tested.

- Advanced Micro Devices (Ref 6) have introduced several upgraded versions of the chips that were used to design and build the MIME. The Am2903 "Super Slice" and the Am2904 Status and Shift Control Unit should be investigated

for possible insertion into the MIME's control processor. Additional speed might be achieved as well as the adding of new features that are available on these chips.

- An additional I/O board could be added that would give the MIME greater access to the outside world. The I/O board could contain additional extensions of the I/O Bus that would interface to additonal RS-232 connectors.

- Additional languages should be emulated, such as the INTEL 8080 or the LSI-11 assembly languages. The execution time that is required on the MIME and that required on the original machine should be compared. This will give an estimator of how good the MIME can emulate another machine's language.

- The different Wright-Patterson Air Force laboratories might be queried for useful projects that would be applicable to the MIME. Many useful projects might be found in this manner.

### Conclusions

Only time and the future can really assess the real benefits of this investigation. Through the indepth use and utilization of the microcode development system the inherent strengths and weaknesses will become apparent.

The microcode development system could be a tremendous tool for the user in his investigations of computer control and micropgramming. The system has enough tools and support programs to allow the user to produce and debug very

complex programs.

The personal rewards of this project were very meaningful. The operation of several different computers were investigated and this has given the author an expanded concept of how computers operate and how they can work together. There were many areas of personal pride and achievements throughout this investigation. The concepts of the development cycle became very apparent and the value of descriptive and useful documentation became paramount in importance. From all aspects this investigation has provided the author with a positive experience in identifying requirements and then producing a workable solution.

It is felt that the objectives of this investigation as stated in Chapter 1 have been accomplished.

## Bibliography

1. Purvis, Richard E. and Ronald D. Yoho. MIME: Microprogrammable Minicomputer Emulator. Unpublished thesis. Wright-Patterson Air Force Base: Air Force Institute of Technology, 1978.
2. Hoyt, Thomas R. and Dean A. Myers. MIME: Microprogrammable Minicomputer Emulator, Volumes I, II, and III. Unpublished thesis. Wright-Patterson Air Force Base: Air Force Institute of Technology, 1979.
3. Digital Microcomputer Handbook, 1977. Digital Microcomputer Processor Handbook, 1979. Digital Memories and Peripherals, 1978. Maynard, Mass.: Digital Equipment Corporation.
4. Croasdale, John and Phil Staubs. Heathkit Line Printer Driver Routine. Unpublished Lab Project. Wright-Patterson Air Force Base: Air Force Institute of Technology, 1980.
5. UCSD PASCAL. Institute for Information Systems. University of California, at San Diego, La Jolla, Ca.
6. The Am2900 Family Data Book with Related Support Circuits. Sunnyvale, CA: Advanced Micro Devices, Inc., 1979.

Vita

Glenn H. Carroll was born on 21 October 1948 in Ogden, Utah. A 1966 graduate of Weber High School, Ogden, Utah, he attended Utah State University for one year. He then attended Brigham Young University, Provo, Utah, from which he received the degree of Bachelor of Science (Electrical Engineering) in December 1973. He received a commission in the USAF through Reserve Officer Training Corps (ROTC) in December 1973, and was assigned to Vandenberg AFB, CA, as a Technical Engineering Team Chief until January 1978. He was then assigned to the Space Shuttle Transportation System as a Launch Processing Development Engineer until entering the School of Engineering, Air Force Institute of Technology, in June, 1979.

Permanent Address: 1035 W. 3925 N.

Ogden, Utah 84404

**DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 59 IS OBSOLETE**

UNCLAS BAGHDAD

SECURITY CLASSIFICATION BY THIS SOURCE: *(When Data Inserted)*

~~UNCLASSIFIED~~

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

also produced: (1) a program to semi-automate the generation of the punched cards used to document the MMIE's hardware configuration, (2) an executive that runs on the LSI-11 to communicate with the other computers and the user, and to transfer the microcode files, and (3) updating the MMIE Modest Monitor to communicate on a higher level and add several new features.

The results provide the user with a very powerful tool in the generation of microprograms. The operational flexibility of the MMIE was increased and gives the user many new options in generating microprograms that will allow him to study microprogramming, emulation, and computer control.

~~UNCLASSIFIED~~

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DA  
FILE

7